ELSEVIER

Theoretical Computer Science 292 (2003) 263–281

Theoretical Computer Science

www.elsevier.com/locate/tcs

Composition sequences for functions over a finite domain

Arto Salomaa

Turku Centre for Computer Science, Lemminkäisenkatu 14A, 20520 Turku, Finland

Abstract

Diverse problems ranging from many-valued logics to finite automata can be expressed as questions concerning compositions of functions over a finite domain. We develop a theory dealing with the depth and complete depth of such functions. Interconnections with synchronizable finite automata are also discussed. Many of the very basic problems turn out to be NP-hard. Also several open problems are pointed out. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Many-valued truth-functions; Length of compositions; Synchronizable automata

1. Introduction: basic definitions

In this paper we will consider functions $g(x)$ whose domain is a fixed finite set N with n elements, $n \geqslant 2$, and whose range is included in N. We will mostly deal with this abstract setup. It is clear that such a setup occurs in many and very diverse situations and interpretations. Depending on the interpretation, different questions will be asked.

The two interpretations we have had mostly in mind are *many-valued logic* and *finite automata*. In the former, the set N consists of n *truth values* and the functions are truth functions. In the latter, the set N consists of the *states* of a finite automaton, whereas each letter of the input alphabet induces a specific function: the next state when reading that letter.

In this paper we will restrict the attention to functions with *one* variable only. In many contexts, especially in many-valued logic, it is natural to consider functions of several variables. Although we want to return to the topic of several variables later, the problems we consider in this paper are naturally formulated for functions of one variable.

We make the following *convention*, valid throughout this paper: n always stands for the number of elements in the basic set N. In most cases we let N simply be the set

E-mail address: asalomaa@utu.fi (A. Salomaa).

PII: S0304-3975(01)00227-4

consisting of the first n natural numbers:

$$N = \{1, 2, \ldots, n\}.$$

Clearly, there are altogether n^n functions in the set N^N we are considering.

Consider a couple of examples. If we are dealing with n-valued logic, and the function g is defined by the equation

$$g(x) = n - x + 1, \quad x = 1, 2, \ldots, n,$$

then g is the well-known *Łukasiewicz negation*. (1 is the truth value "true", n is the truth value "false", whereas the other numbers represent the intermediate truth values.) If we are dealing with a finite deterministic automaton whose state set equals N, the function g defined by the equation above could be viewed as transitions affected by a specific input letter a. Under this interpretation, the letter a interchanges the states n and 1, the states $n-1$ and 2, and so forth. Whether or not there is a *loop* affected by the letter a, that is, whether or not some state is mapped into itself, depends on the parity of n.

When we speak of "functions", without further specifications, we always mean functions in the setup defined above. Clearly, the *composition* ab of two functions a and b is again a function. We read compositions from *left to right*: first a, then b. This is in accordance of reading the input words of a finite deterministic automaton from left to right. Because of this convention, it is natural to write the argument x of a function to the left: $(x)ab = ((x)a)b$. Observe that gg equals the identity function for the Łukasiewicz negation g.

Our point of departure will be a nonempty set $\mathbf{F}$ of functions. The only assumption about the set $\mathbf{F}$ is that it is a nonempty subset of the set N^N of all functions; $\mathbf{F}$ may consist of one function or of all functions. We will consider the set $\mathbf{G}(\mathbf{F})$ of all functions *generated* by $\mathbf{F}$, that is, obtained as compositions (with arbitrarily many composition factors) of functions from $\mathbf{F}$. If a particular function f can be expressed as a composition of functions a_i, $i = 1, 2, \ldots, k$, belonging to $\mathbf{F}$:

$$f = a_1 a_2 \ldots a_k,$$

where some of the functions a_i may coincide, then the word $a_1 a_2 \ldots a_k$ is referred to as a *composition sequence* for f, denoted $cs(f)$. (In this brief notation we assume that the set $\mathbf{F}$ is understood.) The number k is referred to as the *length* of the composition sequence, in symbols, $|cs(f)| = k$. The function f is often referred to as the *target function*. Observe that our composition sequences have to be nonempty, implying that the identity function is not necessarily in $\mathbf{G}(\mathbf{F})$; it is in there exactly in case the set $\mathbf{F}$ contains at least one permutation.

Clearly, $\mathbf{G}(\mathbf{F})$ can be viewed as the *semigroup* generated by $\mathbf{F}$. However, we will prefer the more straightforward approach and will not use semigroup-theoretic terminology in the sequel.

The set $\mathbf{F}$ is termed *complete* if all of the n^n functions are in $\mathbf{G}(\mathbf{F})$. Following [12], we will speak also of the *genus* and *type* of a function f. A function f is said to be

of *genus* t if it assumes exactly t values. Thus, the genus equals the cardinality of the range of f. A function f of genus t is said to be of *type* $m_1 \oplus m_2 \oplus \cdots \oplus m_t$, where $m_1 + m_2 + \cdots + m_t = n$ if, for each i with $1 \leqslant i \leqslant t$, there is a number y_i such that f assumes y_i as a value exactly m_i times. Obviously, we do not change the type if we change the order of the numbers m_i, which means that the operation $\oplus$ is commutative. For instance, permutations are of genus n and of type $1 \oplus 1 \oplus \cdots \oplus 1$. The type of a function f tells us how many values f assumes and how many times it assumes each value. It does not tell us what these values are and in what order they are assumed. The numbers m_i are referred to as the *terms* of the type.

Since n is finite, a specific function f can always be defined by a table. Omitting the argument values, this amounts to giving the *value sequence* of f, that is, the sequence $f(1), f(2), \ldots, f(n)$ of its values for the increasing values of the argument. The Łukasiewicz negation can be defined in this way by its value sequence

$$n,\ n-1, \ldots, 2,\ 1.$$

When there is no danger of confusion, we omit the commas from the value sequence. Thus, for $n = 6$, the value sequence of the Łukasiewicz negation reads 654321.

A brief description about the contents of this paper follows. Basic facts concerning completeness are presented in Section 2. Section 3 presents a case study, $n = 3$, emphasizing phenomena of interest for the subsequent considerations. Section 4 introduces notions of our main concern in this paper, dealing with the length of composition sequences. General observations about long and short composition sequences are presented. Although it is clear a priori that problems in our setup will be decidable, most of the central problems are intractable, NP-hard. This will be shown in Section 5 for the minimality problem of composition sequences, as well as for the problem of membership in sets $\mathbf{G}(\mathbf{F})$. Section 6 discusses some specialities of constant functions. It turns out that the successive ranges obtained from a composition sequence are of particular interest in this case. The final section deals with finite deterministic automata, especially *synchronizable* ones. The subject matter falls within the theory of constant functions.

2. Completeness

We begin with a straightforward observation. However, the observation turns out to be very important in diverse considerations concerning composition sequences.

Lemma 1. *The type of a composition ab is obtained from the type of a in such a way that each term in the type of ab is written as the sum of zero or more terms in the type of a, in which process each term in the type of a has to be used exactly once. Consequently, the genus of ab equals at most that of a.*

Proof. The second sentence is an immediate consequence of the first. (Observe also that, by the definition, the genus of ab cannot exceed that of b.) To prove the first sentence, assume that a is of the type $m_1 \oplus m_2 \oplus \cdots \oplus m_k$. This means that, for each i with $1 \leqslant i \leqslant k$, there is a number y_i such that a assumes y_i as its value exactly m_i times. If b permutes the numbers y_i, then the type of ab equals that of a. Otherwise, b identifies some of the numbers y_i, in which case the type of ab is obtained by the summing operaton described in the lemma. □

Lemma 1 can often be used to show that a composition sequence cannot any more be continued to yield a given target function. For instance, if the terms of the type of a composition sequence are all even, then the sequence cannot be continued to yield a target function whose type contains an odd term. Similarly, a composition sequence of type $2 \oplus 3$ cannot be continued to yield a target function of type $1 \oplus 4$. On the other hand, if the target function is a constant i and if it is reached by a composition sequence w (where w is a word over the alphabet of the names of the functions in **F**), then it is reached also by every composition sequence xw, where x is a word over the same alphabet. Moreover, every composition sequence xwx_1 yields a constant but not necessarily the constant i.

We now consider conditions for a set **F** to be complete. The proof of the following characterization, [13], gives also some ideas about the possible lengths of composition sequences. (The origins of the result come from many sources, for instance, see [15, 8] and their references.) We exclude case $n = 2$, for which the two functions with the value sequences 21 and 11 form a complete set.

Theorem 1. *Assume that $n \geqslant 3$. Then three functions generate all functions if and only if two of them generate the symmetric group S_n and the third is of genus $n-1$. No less than three functions generate all functions.*

Proof. Let a and b be permutations forming a basis of the symmetric group S_n and let c be a function of genus $n-1$. We denote by F_i, $1 \leqslant i \leqslant n$, the set of all functions of genus i. By the choice of a and b, the set F_n is generated by them. Proceeding inductively, we assume that every function in the set F_i, $1 < i \leqslant n$ is generated by the three given functions a, b and c.

Let f be of genus $i-1$. There are two distinct numbers p and q with $(p)f = (q)f$. Moreover, there is a number r such that $(x)f \neq r$, for all x. Let g be the function defined by

$$(x)g = (x)f \quad \text{for } x \neq p, \ (p)g = r.$$

The function g being of genus i, it is generated by the three given functions, according to the inductive hypothesis.

Since the function c is of genus $n-1$, there are two distinct numbers k and l such that $(k)c = (l)c$ and, moreover, the values $(x)c$ with $x \neq l$ are all different. Furthermore, there is a number u (in the basic set N) such that c does not assume the value u.

Let d_1 be the function which maps the number $(k)c=(l)c$ to k, the number u to l, and the number $(x)c$ to x whenever $x\neq k,l$. Clearly, d_1 is a permutation and, therefore, the function $c_1=cd_1$ is generated by the three given functions. We observe that $(l)c_1=k$, whereas $(x)c_1=x$ for $x\neq l$.

Finally, let d_2 be any permutation mapping r to l and $(p)f$ to k. Then it is easily verified that

$$f=gd_2c_1d_2^{-1},$$

which completes the induction.

Since the symmetric group S_n is not cyclic, the rest of the theorem follows by Lemma 1: No less than three functions suffice, and whenever three functions are generators, two of them constitute a basis for S_n and the third is of genus $n-1$. □

Suppose you want to find a composition sequence for a function f in terms of the three functions a,b,c given in Theorem 1 and, moreover, you want the composition sequence to be *as short as possible.* We will see that no polynomial bound, in terms of n, can be given for the length of such minimal composition sequences. Otherwise, very little can be said in general. Although there are comprehensive studies concerning bases for the symmetric group (see, for instance, [8]), the lengths of compositions arising from a given basis have not been studied very much. We consider a simple example.

Assume that $n=6$ and that the functions a,b,c of Theorem 1 are defined by the value sequences 213456, 234561 and 112345, respectively. Thus, a is transposition (12), b is the circular permutation (123456), whereas c is of genus 5 and maps 1 to itself and all the other numbers to the preceding number. The target function f is defined by the value sequence 311344 and, consequently, is of type $2\oplus 2\oplus 2$. Roughly following the proof of Theorem 1, the composition sequence (perhaps not the shortest one)

$$b^4d^2cdb^4db^2d^3bc^2db^4d^2b^3d^2b^2d^4ac^2db^4d^2b^3d^2$$

can be given for f, where we have abbreviated $d=ab$. The length of this composition sequence is 75.

According to a result of Piccard [8], given any nonidentical permutation in S_n, another permutation can be effectively constructed such that the two permutations form a basis of S_n. The case $n=4$ is exceptional because a permutation in the Klein Four-Group cannot be extended to a basis of S_4. Thus, we obtain the following corollary of Theorem 1.

Theorem 2. *Assume that $n\neq 4$. Given a nonidentical function a of genus n and a function c of genus $n-1$, a function b can be effectively constructed such that the set $\{a,b,c\}$ is complete.*

It should be emphasized, finally, that questions of completeness are of central importance in the theory of many-valued logics. However, in this theory, it is quite essential

to consider functions of arbitrarily many variables rather than functions of only one variable, as we have done. Composition sequences will then be replaced by composition trees. Functions of two variables are easily constructed such that one function generates the three functions a, b, and c of Theorem 1. Indeed, such a function generates all functions (of arbitrarily many variables) and is customarily referred to as a *Sheffer function.* More information is contained in [13, 14]. Thus, a Sheffer function constitutes alone a complete set. Completeness is also intimately connected with axiomatizability.

3. A case study

We now consider the special case $n=3$. Since there are altogether only 27 functions, the situation is easily handled. However, this case serves as a good illustration of many of the basic phenomena. We define the functions in a complete set $\{a,b,c\}$ by the value sequences 231, 132 and 223, respectively. Thus, a is the circular permutation (123), b is transposition (23), whereas g is of genus 2 and maps 1 to 2 but keeps 2 and 3 fixed.

Table 1 lists all of the 27 functions, giving in each case the value sequence and a shortest possible composition sequence.

Thus, altogether 10 different functions are represented by words of length $\leqslant 2$. Additionally, 6 functions are represented by words of length 3, and 6 further functions by words of length 4. The remaining exceptional functions $1,5,10,19,27$ require a longer word for their representation.

As regards constants, we observe that one of them is represented by a word of length 4 but by no shorter words. The composition sequence ca^2ca^2 for the constant 1 is of special interest for our subsequent discussions. Reading the sequence from left to right, consider the range of the function obtained so far. When c is applied to the whole

Table 1

Name	Values	Composition		Name	Values	Composition
1	111	ca^2ca^2	•	2	112	ca^2
3	113	cba	•	4	121	aca^2
5	122	a^2cab	•	6	123	b^2
7	131	$acba$	•	8	132	b
9	133	a^2ca	•	10	211	a^2ca^2
11	212	$acab$	•	12	213	ba
13	221	cab	•	14	222	ca^2c
15	223	c	•	16	231	a
17	232	ac	•	18	233	a^2cb
19	311	$abcba$	•	20	312	a^2
21	313	aca	•	22	321	ab
23	322	a^2c	•	24	323	acb
25	331	ca	•	26	332	cb
27	333	ca^2ca	•			

set $N=\{1,2,3\}$, we get the range $\{2,3\}$. When a is applied to the latter, we get the range $\{1,3\}$, and so forth. Altogether we get the sequence of ranges

$$\{1,2,3\},\ \{2,3\},\ \{1,3\},\ \{1,2\},\ \{2\},\ \{3\},\ \{1\}.$$

It has no repetitions and contains all nonempty subsets of N. In the sequel, we will express this by saying that the composition sequence ca^2ca^2 is both *range-reduced* and *range-complete*. Each of the words

$$cabca^2,\ cabcba \text{ and } ca^2cba$$

has this same property.

Since there are altogether 120 nonempty words of length $\leqslant 4$, some functions possess many representations using such words. The greatest number is possessed by the function 15 which has no less than 17 such representations:

$$c, c^2, bac, b^2c, cb^2, c^3, a^3c, a^2bc, bac^2, b^2c^2, ca^3, cacb, cbac, cb^2c, cbcb, c^2b^2, c^4.$$

This case was studied also in [6].

4. Depth of functions

Returning to our basic setup, we consider a set $\mathbf{F}$ of functions. That a function f is generated (or represented) by $\mathbf{F}$ means that a composition sequence can be given for f. A composition sequence can be viewed as a *word* over the alphabet $\mathbf{F}$. (We do not distinguish here functions from their names and use the same notation $\mathbf{F}$ also for the alphabet.) We denote by

$$L(\mathbf{F}, f)$$

the set of all composition sequences for f, that is, the *language* over the alphabet $\mathbf{F}$ whose words, viewed as composition sequences, yield the function f. Clearly, the language $L(\mathbf{F}, f)$ can be empty (this is the case when f cannot be expressed as a composition of functions in $\mathbf{F}$) or infinite (composition sequences may contain redundant parts and be arbitrarily long). However, the following lemma is rather obvious.

Lemma 2. *The language $L(\mathbf{F}, f)$ is regular, for any $\mathbf{F}$ and f.*

Proof. For each i, $i=1,\ldots,n$, consider the finite deterministic automaton A_i with the state set N, input alphabet $\mathbf{F}$, initial state i, final state set $\{f(i)\}$ and transition function δ, where $\delta(j,a)=(j)a$, for all states j and input letters a. Clearly, $L(\mathbf{F}, f)$ consists of all nonempty words in the intersection of the languages accepted by the automata A_i, where i ranges from 1 to n. □

The automaton corresponding to the complete set **F** discussed in Section 3 is defined by the following transition table. (Initial and final states have to be specified for each particular case.)

δ	a	b	c
1	2	1	2
2	3	3	2
3	1	2	3

We now come to the central notions concerning the length of composition sequences. For any language L, we denote by $\min(L)$ the length of the shortest word in L. (If L is empty, we agree that $\min(L)=\infty$.) The *depth* of a function f *with respect* to the set **F**, in symbols $D(\mathbf{F}, f)$, is defined by the equation

$$D(\mathbf{F}, f) = \min(L(\mathbf{F}, f)).$$

Thus, the depth of a function with respect to a particular set can also be ∞.

The *depth* of a function f is defined by the equation

$$D(f) = \max(D(\mathbf{F}, f)),$$

where **F** ranges over all sets with the property

$$L(\mathbf{F}, f) \neq \emptyset.$$

Because, for any f, there are sets **F** with this property, we conclude that the depth of a function is always a positive integer. (The notion of depth was introduced in [6], where it was referred to as "complexity".)

Given a set **F** and a function f, a composition sequence w for f is referred to as *minimal* if its length satisfies the equation

$$|w| = D(\mathbf{F}, f).$$

Clearly, a function can possess several minimal composition sequences. The depth $D(f)$ tells how long a composition sequence can be in the worst case. If a composition sequence w for f can be written in the form $w = w_1 w_2 w_3$, where the sequences w_1 and $w_1 w_2$ define the same function, then also $w_1 w_3$ is a composition sequence for f and, consequently, the original sequence w is not minimal. Since the total number of functions is n^n, we get the upper bound

$$D(f) \leqslant n^n \quad \text{for any } f.$$

The following result shows that no polynomial upper bound (in terms of n) can be obtained. The proof uses an idea applied also in the discussion of the payoff for the transition from a nondeterministic to a deterministic finite automaton.

Theorem 3. *There is no polynomial $P(n)$ such that $D(f) \leqslant P(n)$ holds for all functions f.*

Proof. Let p_i be the ith prime, and consider numbers n of the form

$$n = p_1 + p_2 + \cdots + p_k.$$

Let a be a permutation in the symmetric group S_n, defined as the product of k cycles of lengths $p_1, p_2, \ldots, p_k$. Let the set $\mathbf{F}$ consist of a only. Let the target function f be the identity function. Clearly,

$$D(\mathbf{F}, f) = p_1 p_2 \cdots p_k = \Pi.$$

By the well-known estimate $p_k \leqslant k^2$, $k > 1$, we obtain $n \leqslant k p_k \leqslant k^3$, whence $k \geqslant \sqrt[3]{n}$. Since obviously $\Pi \geqslant k!$, we obtain finally

$$D(f) \geqslant [\sqrt[3]{n}]!$$

which establishes the claim. □

The proof shows also that there are specific functions having no polynomial upper bound for their depth. The method is quite general: instead of the identity function we can choose, for instance, the function mapping each element in a cycle to the preceding element.

The *complete depth* $D_C(f)$ of a function f is defined also by the equation

$$D_C(f) = \max(D(\mathbf{F}, f)),$$

but now $\mathbf{F}$ ranges over *complete* sets of functions. Hence, it is a priori clear that $L(\mathbf{F}, f) \neq \emptyset$.

It follows by the definition that every function f satisfies

$$D_C(f) \leqslant D(f).$$

However, lower bounds such as the one given for $D(f)$ in the proof of Theorem 3 are much harder to obtain for $D_C(f)$, for the simple reason that we have much less leeway if we have to restrict the attention to complete sets $\mathbf{F}$ only. On the other hand, we do not know examples of functions for which the above inequality is strict. Such functions do not exist if $n = 2$, and also probably not for $n = 3$. It still seems that the following conjecture holds.

Conjecture 1. *Assume that $n \geqslant 4$. Then there is a function f with the property*

$$D_C(f) < D(f).$$

We already defined the notion of a *minimal* composition sequence. A related notion is that of a *reduced* composition sequence. We again consider a given set $\mathbf{F}$ of functions. By definition, a composition sequence w for a function f, $cs(f) = w$, is *reduced* if

$cs(f) = w'$ holds for no word w', obtained from w by removing some letters. (In other words, w' is a nonempty scattered subword of w distinct from w.)

Clearly, a minimal composition sequence is always reduced. The converse does not necessarily hold, many examples can be obtained from Section 3. For instance, consider function 6. The composition sequence $cs(6) = a^3$ is reduced but not minimal because also b^2 is a composition sequence for function 6. A composition sequence w is not reduced if it can be written in the form $w = w_1w_2w_3$, where the sequences w_1 and w_1w_2 define the same function.

Another related notion is that of a *range-reduced* composition sequence. We already mentioned this notion in Section 3 and will return to it later on in Section 6.

Theorem 3 shows that no reasonable upper bounds are obtainable in the general case, that is, for arbitrary functions. Theorem 3 shows also that some specific functions have intractably long composition sequences in the worst case. However, the situation is different for some other specific functions, notably *constant* functions. This is an area widely studied in the past (see, for instance, [1–4, 9–11] and their references). The area is closely linked to *synchronizable* finite automata and the so-called *Černý Conjecture*. We will return to these matters in Sections 6 and 7.

We will now establish a cubic upper bound for the depth of any constant function. Similar bounds have been obtained, for instance, in [3, 9] in a different setup and by different methods.

Theorem 4. *Every constant function f_c satisfies the inequality*

$$D(f_c) \leqslant n^3/2 - 3n^2/2 + 2n.$$

Proof. We consider an arbitrary set $\mathbf{F}$ and a constant function f_c in $\mathbf{G}(\mathbf{F})$. We have to construct a composition sequence w_c for f_c such that

$$|w_c| \leqslant n^3/2 - 3n^2/2 + 2n.$$

Necessarily, $\mathbf{F}$ contains a function a merging two numbers:

$$(i)a = (j)a, \quad i \neq j.$$

Pairs (i, j) mergeable in this sense by a function in $\mathbf{F}$ are referred to as *critical.* To obtain w_c, we make use of critical pairs whenever possible and reduce the genus of the resulting function (recall Lemma 1), until the genus 1 is reached. We need at most $n-1$ genus-reduction steps. When the genus 1 is reached, we might still need another $n-1$ steps until the target constant f_c is reached. How many steps do we need after reaching genus $k, n-1 \geqslant k \geqslant 2$, before the next reduction can be accomplished?

In this case, we know that a critical pair (i, j) will eventually be reached, that is, it appears in the range of the function constructed so far. Thus, we have a sequence of ranges $M_1, \ldots, M_t$, each of cardinality k. The range M_1 results from the preceding

reduction and M_t contains the critical pair (i,j). We denote $(i,j)=(i_t,j_t)$, and consider the "history"

$$(i_1,j_1),\ldots,(i_{t-1},j_{t-1}),(i_t,j_t)$$

of the critical pair (i,j), that is, the sequence of pairs (i_v,j_v) in the sets M_v giving rise to the critical pair (i_t,j_t) in M_t. In other words, for some functions b_v in $\mathbf{F}$,

$$(i_v)b_v = i_{v+1}, \quad (j_v)b_v = j_{v+1}, \qquad v = 1,\ldots,t-1.$$

The following observation is crucial. If there are two identical (unordered) pairs in the history, the latter one of them and the intermediate part can be removed. The resulting sequence of pairs is still a history of the critical pair, and the corresponding composition sequence a composition sequence for f_c. Consequently, the answer to our question above is: we need at most as many steps as is the number of unordered pairs, subtracted by 1.

Collecting the results, we obtain the upper bound

$$(n-2)(n(n-1)/2\ -1)+2(n-1)$$

for the length of w_c. But this coincides with the upper bound claimed in the theorem. □

By Section 3, the upper bound obtained (6) is the best possible for $n=3$. Moreover, in this case $D_C(f_c)=D(f_c)=6$. For larger values of n, one can do considerably better than in the above proof when the genus is high [10]. There are also various other methods of improving the cubic polynomial. However, no general results about reducing its degree have been obtained.

5. NP-hard problems

It is quite obvious that, due to finite upper bounds, all reasonable problems are *decidable* in our setup. For instance, given a function f and a set set $\mathbf{F}$, we can decide whether or not f is in $\mathbf{G}(\mathbf{F})$. This follows because of the trivial upper bound n^n for the length of minimal composition sequences. We can also test, by trying out all of the finitely many possibilities, whether or not a given composition sequence for f is minimal or reduced.

However, as far as complexity issues are concerned, practically all of the basic problems are *intractable.* In this section we will present some results in this direction.

We will speak also of *classes* of functions. By a *class* we mean the set of all functions satisfying certain (finitely many) defining relations. For instance, the relations

$$(1)g = 1, \quad (2)g = 2$$

define the class of functions keeping values 1 and 2 fixed, and the relations

$$(1)g = (2)g = \cdots = (n)g$$

(finitely many for any specific n) define the class of all constant functions. By the *class membership problem* we understand the problem of deciding, for a given **F** and a class C of functions, whether or not **G**(**F**) contains some function belonging to the class C.

Observe that **G**(**F**) is not, in general, a class in our sense. Observe also that, from the complexity point of view, we have to vary also **F** because, otherwise, everything happens within a fixed finite bound.

By the *minimality problem* we understand the problem of deciding, for a given **F** and a composition sequence w in terms of **F**, whether or not w is minimal.

Theorem 5. *The class membership problem is* NP*-hard.*

Proof. We apply reduction to SAT, the satisfiability problem for propositional formulas in conjunctive normal form. We are using SAT but the same argument applies also to problem 3-SAT, where each of the disjunctive clauses contains only three terms.

Assume we are given a propositional formula α in conjunctive normal form, having k variables $x_1,\ldots,x_k$ and l clauses of disjunctions α_i, $i=1,\ldots,l$. Thus,

$$\alpha = \alpha_1 \wedge \alpha_2 \wedge \cdots \wedge \alpha_l.$$

We now choose $n=kl+2$ and arrange the numbers $1,\ldots,kl$ in an array as follows:

$$\begin{array}{ccc} 1 & 2 & \cdots l \\ l+1 & l+2 & \cdots 2l \\ \vdots & \vdots & \vdots \\ (k-1)l+1 & (k-1)l+2 & \cdots kl \end{array}$$

We consider the set **F** consisting of two functions a and b, defined as follows. The numbers n and $n-1$ are "sinks" for both functions:

$$(n-1)a = (n-1)b = n-1, \quad (n)a = (n)b = n.$$

For the numbers in the above array, that is, for the numbers up to $n-2$, the two functions are defined as follows. Consider the number $(i-1)l+j=u(i,j)$ in the position (i,j) in the array. If the variable x_i appears unnegated (resp. negated) in α_j, then a (resp. b) maps $u(i,j)$ to n. In all other cases both a and b map $u(i,j)$ to the next element $u(i+1,j)$ in the same column, except that the elements $u(k,j)$ in the last row are mapped into $n-1$ by both a and b.

Finally, we define the class C to consist of all functions g satisfying

$$g(1) = g(2) = \cdots = g(l) = n.$$

We claim that **G**(**F**) contains a function in C exactly in case α is satisfiable.

Observe first that the columns in our array correspond to the clauses and the rows to the variables of α. Every composition sequence of length $\geqslant k$ maps every number to one of the two numbers $n-1$ and n.

Assume first that $t_1, t_2, \ldots, t_k$ is a truth value assignment for the variables $x_1, x_2, \ldots, x_k$ satisfying α. Let $c_1 c_2 \cdots c_k = w$ be the composition sequence such that $c_i = a$ (resp. $c_i = b$) if t_i is the truth value "true" (resp. "false"), for $i = 1, 2, \ldots, k$. Consider any number j, $1 \leqslant j \leqslant l$. Let x_i be a variable (there may be several of them) in the clause α_j satisfying α_j. (Thus, t_i is "true" or "false" according as x_i appears in α_j unnegated or negated.) By the definition of a and b, and by the choice of w,

$$(j)c_1 \ldots c_i = n = (j)w.$$

Because j was arbitrary, w defines a function in C.

Conversely, assume that the composition sequence w defines a function in C. Then also the prefix of w of length k,

$$u = d_1 d_2 \cdots d_k$$

defines a function in C. We now contruct a truth value assignment $t_1, t_2, \ldots, t_k$ such that t_i is "true" (resp. "false") if $d_i = a$ (resp. $d_i = b$), for $1 \leqslant i \leqslant k$. Consider an arbitrary clause α_j. We know that $(j)u = n$. Consequently, for some i, $(j)d_1 \cdots d_i = n$. We choose the smallest such i and conclude that the assignment t_i for x_i satisfies α_j. Since again j was arbitrary, the constructed truth value assignment satisfies α. We have carried out the proof in both directions, and our claim follows. □

We still illustrate the construction with the following example. Consider the following propositional formula α in 3-conjunctive normal form:

$$\begin{aligned}\alpha = {} & (\sim x_1 \vee \sim x_2 \vee \sim x_3) \wedge (\sim x_1 \vee x_2 \vee \sim x_3) \wedge (x_1 \vee x_2 \vee \sim x_4) \\ & \wedge (\sim x_1 \vee x_2 \vee x_4) \wedge (x_1 \vee x_2 \vee \sim x_5) \wedge (x_1 \vee \sim x_2 \vee \sim x_5) \\ & \wedge (x_1 \vee x_3 \vee x_4) \wedge (\sim x_1 \vee x_3 \vee \sim x_5) \wedge (x_1 \vee \sim x_4 \vee x_5) \\ & \wedge (x_2 \vee \sim x_3 \vee x_4) \wedge (x_3 \vee x_4 \vee x_5) \wedge (x_3 \vee \sim x_4 \vee x_5).\end{aligned}$$

Thus, there are 5 variables and altogether 12 clauses. Following our construction, we get $n = 62$.

The subsequent array is written as in the proof. We also indicate the numbers mapped by a or b to the number 62:

1b	2b	3a	4b	5a	6a	7a	8b	9a	10	11	12
13b	14a	15a	16a	17a	18b	19	20	21	22a	23	24
25b	26b	27	28	29	30	31a	32a	33	34b	35a	36a
37	38	39b	40a	41	42	43a	44	45b	46a	47a	48b
49	50	51	52	53b	54b	55	56b	57a	58	59a	60a

Thus, the array specifies completely the functions a and b. We see that, for instance, $(13)b = (31)a = 62$. Apart from the specified values, every number is mapped by both a and b to the next number in the same column, except that the numbers in the last row are mapped to 61. The numbers 61 and 62 are "sinks".

It turns out that the word *baabb* (or any word with the prefix *baabb*) defines a function mapping each of the numbers from 1 to 12 into the number 62 and, consequently, this function is in C. The number 6 is mapped to 62 only by the last letter of this word, because we have $(6)baab = 54$. This word is the only possible, and gives rise to the truth value assignment, where the variables x_2 and x_3 get the value "true", and the other three variables the value "false". This is the only assignment satisfying the formula.

Theorem 6. *The minimality problem is* NP-*hard.*

Proof. The minimality problem amounts to the problem of deciding whether or not a given function has a composition sequence of a given length. We apply reduction to the satisfiability problem, the construction being almost the same as in the preceding proof.

Given a propositional formula α, with k variables and l clauses, we again consider $n = kl + 2$ and define the two functions a and b almost as before. The only difference is that now the value $n - 1$ is not a "sink" but rather $(n - 1)a = (n - 1)b = n$. Our target function is the constant n. Clearly, it is defined by a composition sequence of length $k + 1$. (In fact, *every* sequence of length $k + 1$ defines it.) But it is defined by a composition sequence of length k exactly in case α is satisfiable. This is shown exactly as in the preceding proof. □

Observe that the target function is a constant in the above proof. Consequently, by Theorem 4, we get the following corollary. The result has been established also in [4].

Theorem 7. *The minimality problem for constants is* NP-*complete.*

6. Constants: further remarks

We spoke already in Section 3 about *range-reduced* and *range-complete* composition sequences. By definition, a composition sequence w is *range-reduced* if it cannot be written in the form $w = w_1w_2w_3$, where the ranges of the functions w_1 and w_1w_2 coincide. (We of course assume that w_2 is not empty.) It is *range-complete* if, for every nonempty subset N_1 of N, there is a prefix (possibly empty or the whole w) w_1 of w such that N_1 is the range of w_1. Because of Lemma 1, a range-complete composition sequence is always a composition sequence for a constant. The two notions concerning ranges are particularly suitable for constants also because of the following lemma.

Lemma 3. *Let w be a composition sequence for a constant. If w is minimal, it is reduced. If w is reduced, it is range-reduced. The converse implications do not hold.*

Proof. Most of the claims follow from the definitions. That a range-reduced composition sequence for a constant is not necessarily reduced is seen from the following

example. Let $n=4$ and let the three functions a,b,c be defined by the value sequences 2341, 2134 and 1231, respectively. Then the composition sequence *caaacaacabc* for the constant 1 is range-reduced but not reduced, because the letter b and the third c can be omitted, and the resulting sequence *caaacaaac* is still a composition sequence for the constant 1. □

While the first implication in the lemma is valid for arbitrary composition sequences, the second implication holds for constants only. For instance, in Section 3 the composition sequence *abcba* for function 19 is reduced but not range-reduced.

The estimate in the following theorem can be obtained from the results in [9] or [4]; the example used is due to [2]. We give a complete proof because we also want to illustrate some other issues important for our considerations.

Theorem 8. *The depth of a constant function* f_c *satisfies* $D(f_c)\geqslant n(n-1)$.

Proof. We consider a set **F** consisting of two functions a and b, where a is the circular permutation $(12\ldots n)$, and b maps n to 1 but keeps the numbers $1,2,\ldots,n-1$ unchanged. Thus, the value sequences of a and b are $2,3,\ldots,n,1$ and $1,2,\ldots,n-1,1$, respectively. We consider the depth of the constant $f_c=n$. Clearly, f_c is generated by the composition sequence $(ba^{n-1})^{n-1}=w$ of length $n(n-1)$. We will prove that f_c is generated by no shorter composition sequence. The proof also shows that the "orthodox" sequence w is the *only* sequence of length $n(n-1)$ generating f_c.

Observe first that any sequence generating f_c must have a prefix generating the constant 1. Moreover, a^{n-1} is the unique shortest sequence yielding the constant n from the constant 1. Thus, we have to show that $(ba^{n-1})^{n-2}b=w_1$ is the unique shortest sequence yielding the constant 1.

Before we do this, let us try some other sequences. Since the aim is to reduce the genus from n to 1, one is tempted to reduce the genus *faster* than what the orthodox sequence does. This makes sense because no backtracking is needed when searching a composition sequence for a constant: if a correct sequence exists at all, any sequence can be continued to yield a correct one. For instance, let $n=8$, giving the value sequences 23456781 and 12345671 for a and b. Then the sequence $(ba^2)^3b$ of length 10 reduces the genus to 4, as opposed to the prefix $(ba^7)^3b$ of length 25 of the orthodox sequence. However, later on one pays the price because the short sequence yields the "bad" range $\{1,3,5,7\}$ of cardinality 4. Indeed, in the continuation altogether length 20 (versus 33 in the orthodox sequence) is needed to get genus 3, length 33 (versus 41) to get genus 2, and length 61 (versus 49) to get genus 1. In the genus 2 one starts with the range $\{1,5\}$ and has to go through all 28 pairs!

We now show that one cannot do better than w. It is convenient to present the argument in the form of a solitaire game. Consider a circle, where n spots at equal distances have been clockwise marked by the numbers $1,\ldots,n$. At the beginning each spot carries a stone. You have two possible moves in this solitaire. In the move a you transfer every stone to the spot lying clockwise next to the preceding spot of the

stone. In the move b you transfer the stone in the spot n, if any, to the spot 1 and remove the stone, if any, from the spot 1. Other stones are left intact in b. (Thus, b just transfers the stone from the spot n to the spot 1 if there is no stone in the spot 1. If there is no stone in the spot n, b leaves everything intact.) The purpose of the solitaire is to reach a situation, where only one stone remains. What is the minimal number of moves for this? Considering how the moves were defined, it is clear that the answer gives the length of the shortest composition sequence for the constant 1.

A *configuration* in our solitaire is a word x of length n over the binary alphabet, where 1 (resp. 0) indicates a position carrying a stone (resp. an empty position). Thus, 1^n is the initial configuration. The *characteristic number* of a configuration x is the length of the longest factor of x consisting of 0's. Here x is viewed circularly, that is, initial and final 0's constitute one factor. Thus the purpose of the solitaire is to increase the characteristic number from 0 to $n-1$. What is the fastest way to do it?

The move a never changes the characteristic number. The move b may increase it. When it does, it always increases it by 1. This happens exactly in case there is a stone in the position n and the longest factor of 0's (or one of the longest factors if there are several of equal length) ends at the position $n-1$. (Under these conditions, the move b increases the characteristic number, no matter whether or not there is a stone in the position 1.) These observations imply that one cannot do better than, always after applying b, move the stone in the position 1, as well as the empty spaces following it, $n-1$ steps ahead by the rule a. (Greedy actions to reduce genus do not increase the characteristic number!) But this gives exactly the composition sequence $(ba^{n-1})^{n-2}b$. □

Conjecture 2 (Černý). *For constant functions* f_c, $D(f_c)=n(n-1)$.

In view of the preceding theorem, one only has to show that the depth satisfies $D(f_c)\leqslant n(n-1)$. The original Černý formulation can be presented in our setup as follows. Whenever $\mathbf{G}(\mathbf{F})$ contains a constant, then the shortest composition sequence for a constant is of length at most $(n-1)^2$. (Thus, one deals with *any* constant rather than a *specified* constant.) Our formulation of the conjecture follows from the Černý formulation but it is conceivable (although most unlikely) that the latter is wrong but our formulation is correct.

Although very old, the Černý Conjecture seems to have drawn little attention in the early years, whereas recently there has been quite much activity around it. (See [1] and its references.) There are numerous results concerning various special cases but very little work about the cases when the upper bound $(n-1)^2$ is actually reached. The set $\mathbf{F}$ in the above proof, and its isomorphic variants, is the only general example known to us. For $n=4$, the two functions with the value sequences 4213 and 1322 constitute a further example [3].

The same questions can be asked for the *complete depth*. If the set $\mathbf{F}$ in the proof of Theorem 8 is augmented by a transposition to make it complete then the constant

1 can be generated by a composition sequence shorter than $(n-1)^2$. However, the reduction is not big.

Conjecture 3. *The complete depth of a constant f_c satisfies*

$$D_C(f_c) \leqslant (n-1)^2 + 2.$$

The conjecture holds for $n=3$ by Section 3. For $n=4$, the bound is reached if the set **F** in the proof of Theorem 8 is augmented by the transposition (23). As regards the Černý formulation, Conjecture 3 gives the upper bound $(n-1)^2-(n-3)$. By Theorem 8, Conjecture 3 implies Conjecture 1.

Instead of considering short composition sequences for constants, one may try to construct long ones that are still range-reduced. In view of the total number of all possible ranges, 2^n-2 is an absolute upper bound. In Section 3 we gave the example $cabca^2$ for the constant 1. For $n=4$, consider the three functions a,b,c defined by the value sequences 2341, 2134, 1231. Then $ca^3ca^2cabca^3$ is a range-reduced composition sequence of length 14 for the constant 4.

The upper bound 2^n-2 is not necessarily reached even if we consider complete sets **F** only. It is reached, for all n, if we may have arbitrarily many functions in **F**: we can then go through all ranges in any prechosen genus-reducing order. The question is much more difficult and very interesting if we can have only a bounded number of functions in **F**. We conjecture that also then it is possible to reach the upper bound.

Conjecture 4. *There is a number k such that, for any n, a set* **F** *of cardinality at most k can be constructed with the property that a constant has a range-reduced composition sequence of length 2^n-2 in terms of* **F**.

7. Synchronizable automata

The classical paper by Moore [7] about *Gedanken experiments* on finite automata, had the general idea to view a finite automaton as a black box and to try to find out some specific facts about it by observing what kind of outputs certain inputs produced. Of course, for each experiment, the overall setup has to be defined explicitly. (See [5] for an early contribution.)

Suppose you know the structure (graph, transition function) of a given finite deterministic automaton A, but do not know the state A is in. How can you get the situation under control? For some automata, not always, there are words, referred to as *synchronizing*, bringing the automaton always to the same state q, no matter from which state you started from. Thus, you first have to feed A a synchronizing word, after which you have the situation completely under control. You can also view the graph of an automaton as a labyrinth, where you are lost. If you then follow the letters of a synchronizing word (and have the global knowledge of the graph of the

automaton), you have found your way. This shows the connection with the well-known *road coloring* problem.

Clearly, a synchronizing word can be viewed as a *composition sequence for a constant*, and we are back in the setup introduced above. Indeed, consider a finite deterministic automaton, without initial and final states, as a pair $(N, \mathbf{F})$, where N is the state set of cardinality n and $\mathbf{F}$ is a set of functions mapping N into N. The set $\mathbf{F}$ determines both the input alphabet and the transition function in the natural way, and input words correspond to compositions of functions. Our convention about reading compositions from left to right is in accordance with the customary way of reading input words from left to right.

An automaton is *synchronizable* if and only if it possesses a synchronizing word. This happens exactly in case a constant function is in $\mathbf{G}(\mathbf{F})$. The Černý Conjecture says that every synchronizable automaton possesses a synchronizing word of length $\leqslant (n-1)^2$. Results presented above can immediately be translated into this automata-theoretic terminology. One can also speak of *complete automata* in the sense that the set $\mathbf{F}$ is complete, and ask (in view of Conjectures 1 and 3) whether synchronizing words for complete automata are shorter.

The proof of Theorem 4 can be converted into a cubic-time algorithm for deciding whether or not a given finite deterministic automaton is synchronizable. However, no (mathematically nice) necessary and sufficient conditions for a finite deterministic automaton to be synchronizable are known.

Our final theorem applies a technique common in many-valued logic. The theorem gives a method of constructing classes of non-synchronizable automata. We say that a function g is *self-conjugate* under a permutation p if it satisfies the equation

$$g = pgp^{-1}.$$

Theorem 9. *Assume that in a finite deterministic automaton $A = (N, \mathbf{F})$ every function in $\mathbf{F}$ is self-conjugate under p, where the permutation p maps no element of N into itself. Then A is not synchronizable.*

Proof. By the assumption, p commutes with every function

$$g \in \mathbf{F} : pg = gp.$$

This implies that p commutes with every function in $\mathbf{G}(\mathbf{F})$. On the other hand, p does not commute with any constant i. This follows because, by our assumption, p maps i to $i_1 \neq i$ and, consequently,

$$pi = i \neq i_1 = ip. \qquad \square$$

As an example, let n be even, $n = 2m$, and let the automaton A have two input letters a and b, where a affects the circular permutation $(12\ldots n)$, and the transitions by b

are defined by

$$(1)b = (2)b = \cdots = (m)b = m + 1, \quad (m + 1)b = (m + 2)b = \cdots = (2m)b = 1.$$

Then A is not synchronizable, although many functions of genus 2 can be expressed as composition sequences in terms of a and b. We can use here the product of transpositions

$$p = (1, m + 1)(2, m + 2) \cdots (m, 2m).$$

References

[1] S. Bogdanović, B. Imreh, M. Ćirić, T. Petković, Directable automata and their generalizations: a survey, Novi Sad J. Math. 29 (1999).

[2] J. Černý, Poznámka k homogénnym experimentom s konečnými automatmi, Mat. fyz. čas SAV 14 (1964) 208–215.

[3] J. Černý, A. Pirická, B. Rosenauerová, On directable automata, Kybernetika 7 (1971) 289–298.

[4] D. Epstein, Reset sequences for monotonic automata, SIAM J. Comput. 19 (1990) 500–510.

[5] S. Ginsburg, On the length of the smallest uniform experiment which distinguishes the terminal states of a machine, J. Assoc. Comput. Mach. 5 (1958) 266–280.

[6] A. Mateescu, A. Salomaa, Many-valued truth functions, Černý's conjecture and road coloring, EATCS Bull. 68 (1999) 134–150.

[7] E.F. Moore, Gedanken experiments on sequential machines, in: C.E. Shannon, J. McCarthy (Eds.), Automata Studies, Princeton University Press, Princeton, 1956, pp. 129–153.

[8] S. Piccard, Sur les bases du groupe symétrique et les couples de substitutions qui engendrent un goupe régulier, Librairie Vuibert, Paris, 1946.

[9] J.-E. Pin, Le problème de la synchronisation, Contribution a l'étude de la conjecture de Černý, Thèse de 3^e cycle a l'Université Pierre et Marie Curie, Paris 6, 1978.

[10] J.-E. Pin, Sur les mots synchronisants dans un automate fini, Elektronische Informationsverarbeitung und Kybernetik EIK 14 (1978) 297–303.

[11] I.K. Rystsov, Reset words for commutative and solvable automata, Theoret. Comput. Sci. 172 (1997) 273–279.

[12] A. Salomaa, A theorem concerning the composition of functions of several variables ranging over a finite set, J. Symbolic Logic 25 (1960) 203–208.

[13] A. Salomaa, On the composition of functions of several variables ranging over a finite set, Ann. Univ. Turkuensis Ser. AI 41 (1960).

[14] A. Salomaa, On basic groups for the set of functions over a finite domain, Ann. Acad. Scient. Fennicae Ser. AI 338 (1963).

[15] N. Vorobév, On symmetric associative systems, Leningrad Gos. Ped. Inst. Uch. Zap. 89 (1953) 161–166 (in Russian).